AMYGDALA $z \mapsto z^2+c$

A Newsletter of Fractals & $\mathcal{M}$ (the Mandelbrot Set)

Copyright © 1986-1994, Rollo Silver

Issue #31-32
August 1994

In This Issue...

This is a double issue, combining #31 and #32 and consists of 24 pages. It inaugurates a series of double issues, followed by #33-34, #35-36, etc. The double issues will run from 20 to 24 pages, and we will endeavor to get out 5 double issues of that size per year.

We will adjust current subscriptions accordingly, to account for the newly expanded size of the issues:

Current Expiration	New Expiration
31	31-32 (current)
32	31-32 (current)
33	33-34
34	33-34
35	35-36
36	35-36
37	37-38
38	37-38

etc.

The slide supplement will no longer be offered or provided as a option. By the time the next issue (#33-34) comes out, I will have alternatives to offer those subscribers who have unfulfilled slide supplement subscriptions.

New or renewing subscriptions will cost $25/five issues (plus postage) instead of ten issues. This should be beneficial to subscribers, since there will be at least the same amount of material for $25, and most likely more.

The lead article in this issue is Daniel Wolf's *Observations on the Method of Synchronous Orbits.* Daniel has adapted Steve Stoft's *Synchronous Orbits* algorithm (SO) into an assembly language program for the Commodore Amiga, and used it to evaluate the effectiveness of the SO algorithm in speeding up fractal iterations.

Ken Shirriff's *Frequently Asked Questions (Part 2)* is an adaptation of the frequently asked questions (FAQ) file for the Usenet newsgroup alt.fractals, where it is posted every two weeks. It is based on the version posted to FRAC-L on September 7, 1992. Part 2 completes the piece begun in the previous issue.

In his article "z^z", Rollo Silver untangles the intricacies of the math involved in iterating with the function $f(z) = z^z$, and discusses problems associated with multivaluedness.

Warut Roonguthai presents an ingenious dodge for creating views consisting of tiny details of the Mandelbrot set using Fractint, when the magnification

CONTENTS

In This Issue... 1
Observations on the Method of Synchronous Orbits 2
Frequently Asked Questions (Part 2) 7
z^z 15
An Ingenious Dodge: Mandelbrots from Julias 16
Fractals in Special Effects for Movies: an E-mail Conversation 16
A Review of FRACTAL PROGRAMMING IN C, by Roger T. Stevens 17
Sculpture of Quaternion Julia Sets: an E-mail Conversation 19
Barnsley's Compression Software: A review 23
Mazes for the Mind: Computers and the Unexpected 23
The Fractal Transform 24
Author 24
Book Reviews Wanted 24
Submitting Articles 24
Circulation 24

required exceeds the limit that Fractint can handle (about 1.56×10^{13}).

Two extracts from the Bitnet traffic on the FRAC-L conference discuss *Sculpture of Quaternion Julia Sets* and *Fractals in Special Effects for Movies.*

We also have reviews of Michael Barnsley's Compression Software for fractals, as well as two important and interesting books dealing with fractals: Cliff Pickover's *Mazes for the Mind,* and Barnsley and Anson's *The Fractal Transform.*

Observations on the Method of Synchronous Orbits

— *Daniel Wolf*

Stoft (Amygdala #26, p.3) describes a method, called Synchronous Orbits (SO) for reducing the time to compute Mandelbrot Set ($\mathcal{M}$-set) images. Stoft provides only a brief and partial description of the method. I decided to experiment with the SO and so developed the implications of Stoft's description into an assembly language program for the Commodore Amiga. As the author of FractalPro for the Amiga, I had an existing body of program code for a fast fractal rendering system to which I could attach the required new modules implementing the SO method. FractalPro also provides an existing basis for speed comparisons, and comparing the program's performance with and without the SO implementation can truly give some indications of the SO method's effectiveness in reducing computation time. Stoft provides only speed comparison data for a given set of image coordinates rendered by three different popular fractal renderers (on IBM/clone platforms), on identical hardware configurations. Stoft's comparisons don't really isolate the impact of the SO method on fractal rendering speed. The purpose of this article is to describe more fully the SO method and to provide some indications of its impact on $\mathcal{M}$-set image rendering times.

The Synchronous Orbit Method

The Mandelbrot Iterative Transform Process

The equation on which images of the $\mathcal{M}$-set are based is familiar to us:

$$z_{k+1} = z_k^2 + z_0 \qquad (1)$$

where z_0 is a point in the complex plane.

The general notion is that we follow a point as its coordinates are repeatedly transformed by squaring and adding. Points whose distance from the origin (during the iterated process of this polynomial transform) never exceed 2 are said to be members of the $\mathcal{M}$-set. Points which 'escape' the circle of radius 2 are usually colored according to the number of times they were transformed prior to the escape. The process of rendering an image is simply to map a computer screen's display pixels to a rectangular array of points in the complex plane. In practice, some arbitrary limit on the number of iterations is set, and those points which fail to escape within that fixed limit of iterations are declared to be members of $\mathcal{M}$.

Neighboring Points

Points which are near neighbors in the complex plane may behave quite differently from each other under this process. Adjacent points on a computer-rendered $\mathcal{M}$-set image correspond to nearly adjacent points in the complex plane. The metric distance between the horizontally adjacent points on a computer display of $\mathcal{M}$ is simply the total width of the complex-plane region under investigation divided by one less than the number of pixels in one row of the display. During the familiar process of isolating smaller and smaller regions of the complex plane to investigate (zooming in), An $\mathcal{M}$-set explorer creates images in which the computer's rectangular grid of points grows finer and finer in the complex plane.

The SO method relies on the continuity of the functions involved: when neighboring points are sufficiently close to each other, they will also be close neighbors after iterative transformation. This hypothesis can't always be depended on: we are, after all, dealing with a fractal object whose very nature is that neighboring points may substantially diverge under iteration. If it is true for a particular (suitably small) region of the complex plane, then we can take advantage of that fact to 'pre-iterate' an entire rectangular region.

Iterating a Rectangular Region

If we follow the motions of a single point under iteration, we see a none-too-well-ordered scatter of locations. If we follow the iteration of an entire rectangular region (and keep track of whether the hypothesis is true of that region) then, though the sequence of locations of the rectangle may not be very orderly, we'd expect to see the rectangle remain rectangular. The SO method relies on the tendency of a very small rectangular region to remain rectangular under iteration. With the SO method, all we iterate are the four corner points of the rectangle. After each round of iterating the rectangle's corner points, we must examine their new locations and check if they've maintained a rectangular relationship. In general the rectangle has a tendency to rotate, enlarge, and deform as it is repeatedly iterated. Figure 1 shows a small square region's corners and their new locations after one round of iterative transformation of each corner point.

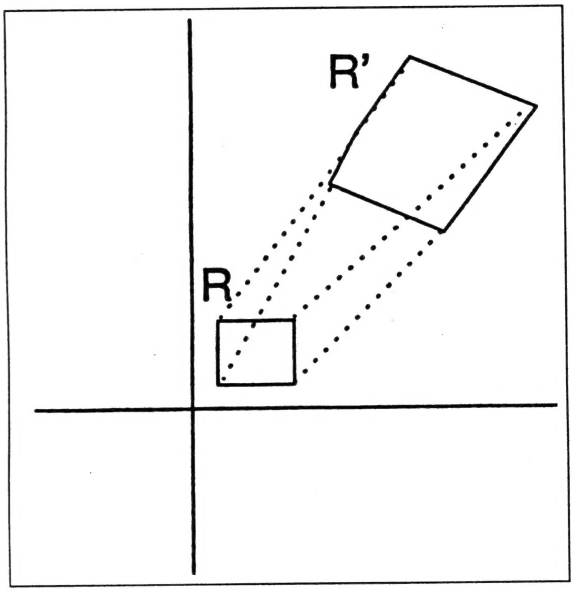

Figure 1. Original rectangular region of complex plane (R) transformed to a new, nearly rectangular region (R') by transforming corner points once.

Substituting the New Rectangle for the Old

For this part of the discussion, we'll only attempt to apply the SO method to 'pre-iterate' a rectangular region once. That will suffice to exemplify the SO method. The implication of a single iteration of a rectangular region all at once is that, during subsequent 'classical' rendering of the image (by repeated transforms of each individual point corresponding to a display pixel), one less iteration need be performed for each point in the entire display. That means that we may be able to use only the iterated locations of the corners of the rectangle to imply the iterated locations of all the other points in the rectangle, and simply to treat the newly transformed rectangle as if it were the original rectangle under consideration for rendering. This is potentially a great time-saving operation when it works.

In general, classical rendering of an image (point-by-point iteration) from an arbitrary rectangular region (with edges not necessarily parallel to the coordinate axes) requires a slightly more complicated rendering routine than one which can always count on rectangles parallel to the axes. This adds a small amount of overhead to the ultimate SO rendering process, which does indeed track point-by-point throughout a rectangle in the classical way.

If we transform the four corner points of a rectangle once, and it is still rectangular (within the tolerance of our display device), we can simply apply the classical point-by-point algorithm for rendering the $\mathcal{M}$-set to the new rectangle (which is albeit no longer parallel to the axes). If the original SO hypothesis is true, then the bottom edge of the original rectangle has simply transformed to a line which is the bottom edge of the new (transformed) rectangle. If 'rectangularity' has been maintained during the transformation, then we can simply plot the $\mathcal{M}$-set image of the new rectangle, adding 1 to the iteration count for each point, since we know that one iteration was already done in the initial transformation of the whole rectangle. We simply begin rendering the newly transformed rectangle in the classical way, moving from point to adjacent point, iterating as we go. At each point, when iteration is complete, add 1 to the iteration count, assign the color value, and plot. The additional complication of rendering the transformed square using the classical method is that we must not for-get that the original rectangle's points must be used for the z_0 values. In other words, the SO method has mapped our original rectangle into a new one, and we must step and track points in that new rectangle to iterate by squaring, but we must step and track the corresponding points in the original rectangle to use in the addition operation of the iteration. This 'double' tracking also adds some overhead to the SO method (see Figure 2).

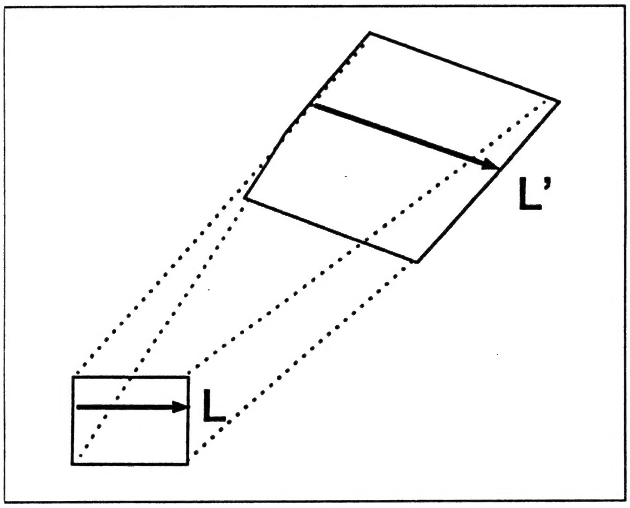

Figure 2. Interpolate points along L' for squaring, but use points along L when adding z_0 in each iteration.

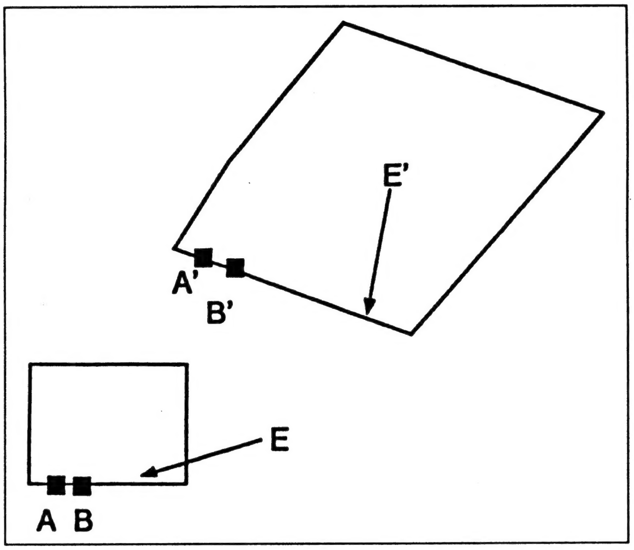

Figure 3. If topological hypothesis holds, if point A iterates to A', and B iterates to B', then interpolating along E' is equivalent to iterating successive points along E.

When we perform the SO series of rectangle transformations, and subsequently apply the 'double-tracked' classical rendering method as necessary, our generalized rectangle renderer interpolates adjacent points on each successive row of our display plot. The coordinates obtained for non-corner points during the final stage of classical rendering of the iterated rectangle should be identical to what we would find had we simply iterated the corresponding points in the original rectangle as many times (see Figure 3). That will not, in general, be absolutely true. A certain amount of interpolative error is introduced by imperfect adherence to the rectangularity criterion, as well as roundoff error. All my calculations were performed using the Motorola 68040's floating-point capabilities in IEEE extended precision (96 bits).

Rectangularity and Interpolation — Verifying the Topological Hypothesis

If the end points of a line (e.g. the bottom edge of the original rectangle) become transformed, then obviously a new line can be drawn between the transformed points. The points on the new line correspond to the points on the original line if the SO topological hypothesis is true (near points transform to near points). In that case the fact that we used only the line's end points to infer the positions of all the points along the transformed line won't matter (see Figure 4). If the topological

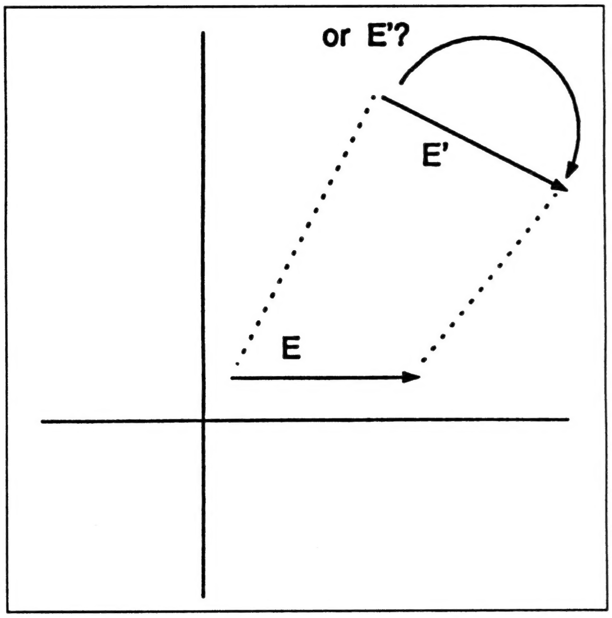

Figure 4. If we only transform endpoints of a line, we can't know if it has been distorted beyond linear interpolability.

hypothesis is true, then the end points provide sufficient information for interpolating the points on the new line accurately. Adjacent points on the transformed line calculated by interpolating between the end points should correspond exactly to the coordinates we'd obtain by transforming corresponding points on the original line. Unfortunately, we can't apply the SO method to individual rows of an $\mathcal{M}$-set rendering. The trouble is that any two endpoints always transform to the endpoints of some other line, but that provides us no evidence that linear interpolation will be valid. If the original line transformed to a curve, we would never know it. Transforming a line doesn't give enough information to let us determine if the topological criterion holds.

If we transform the corner points of a rectangle, though, we can make some measurements and see just how rectangular it remains after transformation. If the region doesn't remain rectangular after transformation, we simply can't use the corners to interpolate points in the transformed region, expecting them to be equivalent to corresponding original points iterated equally, and expect to get an accurate rendering.

Stoft did not state his rectangularity criterion. I chose a simplified rectangularity test: comparing the two differences between the *x*-positions of the endpoints of the top and bottom edges of the region after iteration. True orthogonality tests might be better, but are costly. Rhomboidal distortion (linear shearing) and curvilinear distortion (non-linearity) are not excluded by my test, but that may not be necessary. As long as the top-edge *x*-distance differs only negligibly from the bottom-edge *x*-distance, the region can be said to satisfy the rectangularity criterion for validity of interpolation. 'Negligible' in this case means that the difference between the two lengths is less than can cause a distorted result (see Figure 5). The rigidity of the rectangularity criterion can be chosen arbitrarily, but a distance difference of less than some fraction of the metric distance between two adjacent points on the display device is sufficient. That simply assures that there is a mapping between points in the original rectangle and those in the transformed rectangle that is linear enough to prevent distortion in the display. After all, if we wish to apply the classical rendering routine to the newly transformed rectangle, it has to be rectangular enough so that we can map its interior points into points on the rectangular display device. The original region is, of course, perfectly rectangular in most programs, since the edges are usually chosen parallel to the coordinate axes. If we assure that we've met the topological criterion, then we can fairly assume that using the corner points to frame a transformed rectangle will let us interpolate among its points in a way which accurately reflects what would have happened had we simply applied the classical rendering method to the original rectangle.

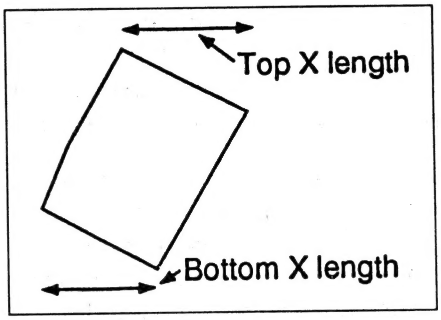

Figure 5. A simple rectangularity test

The Extended SO Method

If you are convinced that we might substitute a rectangular region created from once-iterated corner points of a true rectangular region for the original rectangular region, you've grasped the chief implication of the SO method.

The SO method's full impact is that we might transform the corners of a rectangular area, check if rectangularity has been maintained, and repeat the process, if it has been. Then we simply apply the classical point-by-point $\mathcal{M}$-set rendering method to the resulting many-fold iterated rectangle, and simply add to the point-by-point count-values the number of times we successfully iterated the corners without losing rectangularity. That latter number represents the number of times all points in the rectangle were iterated together as a region. The extended SO method means that we might be able greatly to reduce the number of iterations required at each point when the loss of rectangularity requires that we finally resort to the classical method on the last successfully rectangular iterated-corners result.

The Recursive Extended SO Method

Stoft notes that once the rectangularity criterion has been violated for a rectangle's corner points under iteration, it is possible to divide the rectangle into quadrants and recursively apply the SO method to the smaller rectangles. In general, we expect that the smaller the rectangle, the further we can iterate its corners and maintain the rectangularity criterion on the resulting new regions.[1] We might expect to gain even greater rendering-time savings by permitting a recursive process to divide the original region repeatedly into smaller and smaller rectangles, as required when a rectangle loses its rectangularity. This recursion ultimately divides the original rectangular region until the rectangles represent only a few pixels. The classical rendering method is applied only to these very small regions.

The overall process here is to first iterate corners of the rectangle covering the whole region being rendered. When it becomes distorted, step back one iteration and subdivide the region into smaller rectangles. iterate those smaller rectangles' corners, and so on. When the smallest region size allowed loses rectangularity, iterate its component points using the classical algorithm, to complete its point-by-point iteration process and end the recursion for that rectangle.

I have not yet explored the impact on rendering times of carrying on the recursion process to smaller and smaller sub-rectangles. The following reports are based on the application of the SO method only to the whole rectangle covering an entire region being rendered.

Is Rectangularity Really Necessary?

The assumption, unstated up to now, is that our rendering routines are only easy to apply to regions in which we can set point-by-point along a row or column of the pixel lattice in the complex plane, or interpolate linearly as corresponding to our rectangular display device. Were we able easily to know the parameters of curvilinear edges of badly deformed (transformed) rectangles, and thus able to interpolate nonlinearly within the arbitrarily shaped transformed region, we could evade the rectangularity criterion. We can't however, evade the topological criterion entirely. If neighboring points fail to transform to neighboring points (a virtual certainty at some point due to the fractal nature of $\mathcal{M}$), then all our interpolations to adjacent points in the transformed region would be suspect, and the end result would be a dramatically distorted image.

Rectangularity appears to be necessary for all $\mathcal{M}$-rending routines which interpolate linearly between adjacent points.

How Well Does SO Work?

The SO method has greatly variable impact on rendering times, which is to be expected. The object in question is a fractal, and we should expect that the tendency to retain rectangularity will vary greatly depending on our choice of original corner points.

Indeed, this is true. For some images, the corners survive numerous iterations without losing rectangularity. Others succumb rapidly to distortion. One benefit of SO is that it is fairly quick. You need only iterate four points repeatedly and do a few simple operations to check rectangularity. The cost of using simple SO is small, and the small amount of additional overhead in the classical rendering routine required to cover arbitrarily-oriented rectangles and double-region tracking has only minor impact.

Imagine an image consisting of an $\mathcal{M}$-set region in which all points have dwell times (count values) between 256 and 512. If we are fortunate enough to find that SO can be applied 128 times before the transformed rectangle becomes badly distorted, then we'll save half the calculation time for those points which would have needed 256 iterations, and we'll save one fourth of the time we'd have needed on the points with dwell of 512. If the dwell times are evenly distributed between 256 and 512 then 128 pre-iterations using SO will save 37.5% of the total rendering time (neglecting the slightly increased time required to do SO).

On one image used frequently as FractalPro's benchmark, I found the rendering time decreased from 127 seconds to 92 seconds: about a 25% improvement. On other images, I've found improvements of from 2% to 10% in rendering time. I point out that SO had no success at all until the width of the original rectangular region was smaller than 10^{-6}. SO achieved even greater success with ever smaller original rectangles. In no case have I observed an improvement of more than 25%. Relaxing the rectangularity criterion led to image distortion ranging from subtle to extreme, with little additional time savings.

---

[1] This is a consequence of the "conformal" nature of the mapping induced by the nth iterate of $z \leftarrow z^2 + c$. The SO method should be applicable to iterations based on any conformal mapping (analytic function), e.g. polynomials, rational functions, and exponential and trigonometric functions.
—Ed.

Stoft's program with recursive SO appeared to be over 100 times faster than its nearest competitor (for one particular image), but we have no idea of how much improvement is due solely to SO in Stoft's results, nor do we know his results on other coordinate regions. The difference between two programs, one written in assembly language and the other written in C or Pascal, could itself account for such a difference. My results are based on comparing performance of a given program with and without simple SO.

Other observations lead me to expect that the overhead of the recursive SO method won't be balanced by additional savings. I made numerous applications of SO to 1/16-sized subregions of some of my experimental pictures to simulate the further subdivision which would be accomplished by recursive SO. I found little increase, if any, in the number of iterations saved by SO on the smaller regions, compared to the original 16x larger regions. Whether most of the 'gain' of SO is accomplished on the region as a whole remains under investigation.

I intend to explore the recursive version of SO, but I speculate that simply applying SO directly to rectangular regions consisting of two rows at a time of the original region will maximize computational returns.[2] Such a two-row rectangle is as small vertically as it can be. Systematic subdivision of the original region can eliminate the costly stacking required in recursive SO with small relative cost.

These remain areas for empirical investigation of computational costs.

Summary

The SO method reported by Stoft is novel. Other methods of improving $\mathcal{M}$-set rending times have been described, but they are either irrelevant (only improving rendering of low magnifications, or of regions with large areas inside $\mathcal{M}$) or counterproductive (with such high computational overhead as to offset any other gains). Stoft's method must be carefully implemented, since it can lead to subtle $\mathcal{M}$-set image distortion due to accumulated interpolation and roundoff errors. I have made observations of the relation between the rectangularity criterion strength and subsequent image quality which point this out clearly. SO does work, and may result in substantial time savings for $\mathcal{M}$-set rendering. Its utility is greatest when applied to very small regions (deep zooms of magnification over 10^9. That is helpful, since deeply-zoomed $\mathcal{M}$-set images often take the longest time to render, and any time savings is welcome. The SO method has been observed to reduce calculation time from 2-25% compared to an identical renderer lacking SO. Further investigations are proceeding.

Reference

- Steven Stoft, *Synchronous Orbits: The Fastest Mandelbrot Algorithm*. Amygdala, #26, page 3.

Frequently Asked Questions (Part 2)

— *Ken Shirriff*

This is an adaptation of the frequently asked questions (FAQ) file for the Usenet newsgroup alt.fractals, where it is posted every two weeks. It is the version posted to FRAC-L on September 7, 1992.

---

[2] I doubt that! The underlying math would suggest subdividing so as to keep the regions as compact as possible. — *Editor.*

The first part of FAQ was published in Amygdala #29.

Subject: The Mandelbrot set

Q6a: What is the Mandelbrot set?

A6a: The Mandelbrot set is the set of all complex c such that iterating $z \leftarrow z^2 + c$ does not go to infinity (starting with $z = 0$).

Q6b: How is the Mandelbrot set actually computed?

A6b: The basic algorithm is:

For each c, start with $z = 0$. Repeat $z \leftarrow z^2 + c$ up to N times, exiting if the magnitude of z gets large.

If you finish the loop, the point is probably inside the Mandelbrot set.

If you exit, the point is outside and can be colored according to how many iterations were completed. You can exit if $|z| > 2$, since if z gets this big it will go to infinity. The maximum number of iterations, N, can be selected as desired, for instance 100. Larger N will give sharper detail but takes longer.

Q6c: Why do you start with $z = 0$?

A6c: Zero is the critical point of $z^2 + c$, that is, a point where $\frac{d}{dz}(z^2 + c) = 0$. If you replace $z^2 + c$ with a different function, the starting value will have to be modified. e.g. for $z \leftarrow z^2 + z + c$, the critical point is given by $2z + 1 = 0$, so start with $-1/2$.

Critical points are important because every attracting cycle for a polynomial or rational function attracts at least one critical point. [P. Fatou]. Thus, testing the critical point shows if there is any stable attractive cycle.

Note that you can precompute the first Mandelbrot iteration by starting with $z = c$ instead of $z = 0$, since $0^2 + c = c$.

Q6d: What are the bounds of the Mandelbrot set? When does it diverge?

A6d: The Mandelbrot set lies within $|c| \leq 2$. If $|z|$ exceeds 2, the z sequence diverges.

Proof: if $|z| > 2$, let $|z| = 2 + a$, where $a > 0$. Then $|z^2 + c| \geq |z^2| - |c| = (2 + a)|z| - |c|$.

If $|z| \geq |c|$, then $2|z| - |c| > |z|$.

So, if $|z| > 2$ and $|z| \geq |c|$, $|z^2 + c| > (1 + a)|z|$, so the sequence diverges. Also, note that $z_1 = c$, so if $|c| > 2$, the sequence diverges.

Q6e: How can I speed up Mandelbrot set generation?

A6e: Look at R. Rojas: *A tutorial on efficient computer graphic representations of the Mandelbrot set*, Computers and Graphics, 1991, (15, 1) 91-100.

Q6f: What is the area of the Mandelbrot set?

A6f: Ewing and Schober computed an area estimate using 240,000 terms of the Laurent series. The result is 1.7274... The behavior of the approximations suggests that the limit is between 1.66 and 1.71. However, the estimates of the area from below, using pixel counting, show that the area is at least 1.52. The large gap between the lower bound 1.52 and the upper bound 1.71 may possibly be an indication that the boundary of the Mandelbrot set has positive area.

Ref: J. H. Ewing and G. Schober, *The area of the Mandelbrot set*, Numer. Math. 61 (1992), 59-72.

Q6g: What can you say about the structure of the Mandelbrot set?

A6g: Just about all you could want to know is in Branner's article in *Chaos and Fractals: The Mathematics Behind the Computer Graphics.*

Q6h: Is the Mandelbrot set connected?

A6h: The Mandelbrot set is simply connected. This follows from a theorem of Douady and Hubbard that there is a conformal isomorphism from the complement of the Mandelbrot set to the complement of the unit disk. (In other words, all equipotential curves are simple closed curves.) It is conjectured that the Mandelbrot set is locally connected, and thus pathwise connected, but this is currently unproved.

Connectedness definitions:

Connected: X is *connected* if there are no proper closed subsets A and B of X such that $A \cup B = X$, but $A \cap B$ is empty. *I.e.* X is connected if it is a single piece.

Simply connected: X is simply connected if it is connected and every closed curve in X can be deformed in X to some constant closed curve. *I.e.* X is simply connected if it has no holes.

Locally connected: X is locally connected if for every point p in X, for every open set U containing p, there is an open set V containing p and contained in the connected component of p in U. *I.e.* X is locally connected if every connected component of every open subset is open in X.

Arcwise (or path) connected: X is arcwise connected if every two points in X can be joined by an arc in X. (Definitions from *Encyclopedic Dictionary of Mathematics*.)

Subject: Julia sets

Q7a: What is the difference between the Mandelbrot set and a Julia set?

A7a: The Mandelbrot set iterates $z^2 + c$ with z starting at 0 and varying c. The Julia set associated with c iterates $z^2 + c$ for fixed c and varying starting z values.

Q7b: What is the connection between the Mandelbrot set and Julia sets?

A7b: Each point c in the Mandelbrot set specifies the geometric structure of the corresponding Julia set. If c is in the Mandelbrot set, the Julia set will be connected. If c is not in the Mandelbrot set, the Julia set will be a Cantor dust (totally disconnected).

Q7c: How is a Julia set actually computed?

A7c: The Julia set can be computed by iteration similar to the Mandelbrot computation. Alternatively, points on the boundary of the Julia set can be computed quickly by using inverse iterations.

Subject: Complex arithmetic and quaternion arithmetic

Q8a: How does complex arithmetic work?

A8a: It works mostly like regular algebra with a couple of additional formulas: (note: In what follows, a, b are reals, x, y are complex, $i = \sqrt{-1}$):

$$i^2 = -1$$

$$e^{a+ib} = e^a(\cos b + i\sin b)$$

From these:

Addition: $(a+ib)\,(c+id) = (a+c) + i(b+d)$

Multiplication:
$(a+ib)\,(c+id) = (ac-bd) + i(ad+bc)$

Division: $\dfrac{a+ib}{c+id} = \dfrac{(a+ib)\,(c-id)}{c^2+d^2}$

Sin: $\sin x = \dfrac{e^{ix} - e^{-ix}}{2i}$

Cos: $\cos x = \dfrac{e^{ix} + e^{-ix}}{2}$

Magnitude: $|a+ib| = \sqrt{a^2+b^2}$

Log: $\log(a+ib) = \log|a+ib| + i\operatorname{atan}\dfrac{b}{a} + 2i\pi n$.
Note that log is multivalued: n can be $0, \pm 1, \pm 2, \ldots$.

Complex powers: $x^y = e^{y\log x}$.

More details can be found in any complex analysis book.

Q8b: How does quaternion arithmetic work?

A8b: Quaternions have four components $(a+ib+jc+kd)$ compared to the two components of complex numbers. Operations such as addition and multiplication can be performed on quaternions, but multiplication is not commutative.

Quaternions satisfy the rules

$$i^2 = j^2 = k^2 = -1,\ ij = -ji = k,\ jk = -kj = i,\ ki = -ik = j.$$

Subject: Iterated function systems

Q9a: What is an iterated function system (IFS)?

A9a: If a fractal is self-similar, you can specify various mappings that map the whole onto the parts. By taking a point and repeatedly applying these mappings you end up with a collection of points on the fractal.

Iterated function systems can be used to make fractal ferns and trees and are also used in fractal image compression.

Fractals Everywhere by Barnsley is mostly about iterated function systems.

Q9b: What is the state of fractal compression?

A9b: According to Barnsley's book *Fractals Everywhere*, this method is based on a measure of deviation between a given image and its approximation by an IFS code. The *Collage Theorem* states that there is a convergent process to minimize this deviation. Unfortunately, according to an article Barnsley wrote for BYTE a few years ago, this convergence was rather slow, about 100 hours on a Cray, unless assisted by a person.

Barnsley *et al* are not divulging any technical information beyond the meager bit in *Fractals Everywhere*. The book explains the idea of IFS codes at length, but is vague about the application of the Collage theorem to specific compression problems.

There is reason to believe that Barnsley's company has *no algorithm* which takes a given reasonable image and achieves the compression ratios initially claimed for their fractal methods. The 1000-to-1 compression advertised was achieved only for a 'rigged' class of images, with human assistance. The best unaided performance I've heard of is good lossy compression of about 80-1.

(This answer is from Tal Kubo <kubo@zariski.harvard.edu> in the comp.compression FAQ file. There is a long list of references there.)

A paper on the algorithm is supposed to have come out this past spring in *IEEE Transactions on Signal Processing*. (More details are welcome.)

Some sort of fractal compression demo program is available by anonymous ftp to lyapunov.ucsd.edu [132.239.86.10] in
/pub/inls-ucsd/fractal-2.0.

Subject: Fractal mountains

Q10: How are fractal mountains generated?

A10: Usually by a method such as taking a triangle, dividing it into 3 subtriangles, and perturbing the center point. This process is then repeated on the subtriangles. This results in a 2-D table of heights, which can then be rendered as a 3-D image.

Subject: Plasma clouds

Q11: What are plasma clouds?

A11: They are a type of FractInt fractal and are similar to fractal mountains. Instead of a 2-D table of heights, the result is a 2-D table of intensities. They are formed by repeatedly subdividing squares.

Subject: Lyapunov fractals

Q12a: Where are the popular Lyapunov fractals described?

A12a: See **Lyapunov Exponent References**, under A12c, below.

Q12b: What are Lyapunov exponents?

A12b: A generalization of an eigenvalue. Given two initial conditions for a chaotic system, a and b, which are close together, the average values obtained in successive iterations for a and b will differ by an exponentially increasing amount. In other words, the two sets of numbers drift apart exponentially. If this is written $e^{n\lambda}$ for n iterations, then e^{λ} is the factor by which the distance between closely related points becomes stretched in one iteration. λ is the Lyapunov exponent. At least one Lyapunov exponent must be positive in a strange attractor. A simple derivation is available in *Deterministic Chaos* by Heinz Georg Schuster, Physics Verlag (1984), p. 18.

Q12c: How can Lyapunov exponents be calculated?

A12c: For the common periodic forcing pictures, the Lyapunov exponent is:

$\lambda = \lim_{N \to \infty} \frac{1}{N} \sum_{n=1}^{N} \log_2 \left| \frac{dx_{n+1}}{dx_n} \right|$. In other words, at each point in the sequence, the derivative of the iterated equation is evaluated. The Lyapunov exponent is the average value of the log of the derivative. If the value is negative, the iteration is stable. Note that summing the logs corresponds to multiplying the derivatives; if the product of the derivatives has magnitude < 1, points will get pulled closer together as they go through the iteration.

Lyapunov Exponent References:

1. *Scientific American* (September 1991).

2. M. Markus, B. Hess. Lyapunov Exponents of the Logistic Map with Periodic Forcing. *Computers and Graphics* 13, 4 (1989): 553-558.

3. Frank, M. and T. Stenges. In *Journal of Economic Surveys* 2 (1988): 103-133.

4. T.S. Parker and L.O. Chua. *Practical Numerical Algorithms for Chaotic Systems.* Springer Verlag (1989): Ch. 3.

Subject: Logistic equation

Q13: What is the logistic equation?

A13: It models animal populations. The equation is $x \leftarrow cx(1-x)$, where x is the population (between 0 and 1) and c is a growth constant.

Iteration of this equation yields the period doubling route to chaos.

For c between 1 and 3, the population will settle to a fixed value. For larger c, the population will oscillate between two values, then four values, eight, sixteen, etc.

For still larger c (between 3.57 and 4), the population behavior is chaotic (as it is for most c values).

Subject: Chaos

Q14: What is chaos?

A14: An attractor is chaotic if at least one of its Lyapunov exponents is positive. Chaos results from the existence of a chaotic attractor.

Chaos is the recurrent behavior of a deterministic dynamical system in which the phase-space divergence of nearby trajectories at an exponential rate results in a limited predictability horizon.

In chaotic iterated systems of the form $x_{i+1} = f(x_i)$, the result after iteration is extremely sensitive to the initial value such that $f^n(x_0 + \varepsilon)$ is nowhere near $f^n(x_0)$.

Chaos results from our inability to predict the future behavior of a deterministic system from initial conditions because of its great sensitivity to initial conditions.

Apparently unpredictable behavior arising in a deterministic system.

Unpredictable yet nonrandom behavior of a system.

Stochastic behavior occurring in a deterministic system.

(Ian Stewart's *Does God Play Dice?*)

Subject: Nonlinearity

Q15: What is nonlinearity? What are nonlinear equations?

A15: Nonlinear maps fail to satisfy the condition that $f(ax + by) = af(x) + bf(y)$, where x and y are vectors, and a and b are scalars. *e.g.* $f(x) = ax$ is linear; $f(x) = x^2$ is nonlinear. Nonlinearity is a map or term that is not linear.

Nonlinear equations are equations containing nonlinearities, *e.g.* $f(x) = ax + x^2$ and $f(x) = ax^2 + bx + c$ are nonlinear.

A nonlinear system gives an output which is not proportional to the corresponding input. Nonlinear dynamical systems possess nonlinear dynamical laws, which are functions of the system's state variables.

In linear systems, $\frac{dy}{dx}$ is a constant, while in nonlinear systems $\frac{dy}{dx}$ = some nonconstant function of x.

Nonlinear equations fail to exhibit linear superimposability. Nonlinear equations can be categorized by differentiability, discontinuity, and 'memory' (e.g. hysteresis in an electric circuit), etc. This can be important to some types of nonlinear analysis such as the Popov hyperstability criterion.

Nonlinearity References:

1. D. Zelinsky. *A First Course in Linear Algebra.* Academic Press (1973): 65.

2. W.A. Brock and E.G. Baek. Some theory of statistical inference for nonlinear science. *Review of Economic Studies* 58,4 (1991): 697-716.

3. John Guckenheimer and Philip Holmes. *Nonlinear Oscillations, Dynamical Systems, and Bifurcations of Vector Fields.* Springer-Verlag (1983).

Subject: What is a fractal?

Q16: What is a fractal? What are some examples of fractals?

A16: A rough or fragmented geometric shape that can be subdivided in parts, each of which is (at least approximately) a reduced-size copy of the whole. (A definition from B. Mandelbrot)

A set of points whose fractal dimension exceeds its topological dimension.

Examples of fractals: Sierpinski triangle, Koch snowflake, Peano curve, Mandelbrot set.

Subject: Fractal dimension

Q17: What is fractal dimension? How is it calculated?

A17: Fractal dimension is the Hausdorff-Besicovitch Dimension.

Roughly, fractal dimension can be calculated by taking the limit of the quotient of the log change in object size and the log change in measurement scale, as the measurement scale approaches zero.

E.g. consider a straight line. Now blow up the line by a factor of two. The line is now twice as long as before. Log 2 / Log 2 = 1, corresponding to dimension 1.

Consider a square. Now blow up the square by a factor of two. The square is now 4 times as large as before (i.e. 4 original squares can be placed on the original square). Log 4 / log 2 = 2, corresponding to dimension 2 for the square.

Consider a snowflake curve formed by repeatedly replacing ——— with —/\—, where each of the 4 new lines is 1/3 the length of the old line.

Blowing up the snowflake curve by a factor of 3 results in a snowflake curve 4 times as large (one of the old snowflake curves can be placed on each of the 4 segments —/\—). Log 4 / log 3 = 1.261... Since the dimension 1.261 is larger than the dimension 1 of the lines making up the curve, the snowflake curve is a fractal.

Fractal Dimension References:

1. Barnsley et. al. *Science of Fractal Images,* Springer Verlag (1985).

2. K.J. Falconer. *The Geometry of Fractal Sets* Cambridge Univ. Press (1985).

3. T.S. Parker and L.O. Chua. *Practical Numerical Algorithms for Chaotic Systems.* Springer Verlag (1989): Ch. 7.

4. J.P. Eckmann and D. Ruelle. In *Reviews of Modern Physics* 57,3 (1985): 617-656.

5. J. Theiler. In *Physical Review A* 41 (1990): 3038-3051.

6. Grassberger and Procaccia. In *Physica D* 9 (1983): 189-208.

References on how to estimate fractal dimension:

1. J. Theiler. Estimating Fractal Dimension. *Journal of the Optical Society of America A: Optics and Image Science* 7, 6 (June 1990): 1055-1073.

2. E. Peters. *Chaos and Order in the Capital Markets.* John Wiley & Sons (1991). ISBN 0-471-53372-6. Discusses methods of computing fractal dimension. Includes several short programs for nonlinear analysis.

Subject: Strange attractors

Q18: What is a strange attractor?

A18: An indecomposable closed invariant set that 'attracts' the points about it which contains a transversal homoclinic orbit. (This orbit accounts for the strangeness.)

The limit set of a chaotic trajectory.

A phase space locus of a bounded long-term dynamical behavior which has a nonzero probability of being observed — its basin of attraction has positive measure — and contains not a smooth manifold structure, but rather a self-similar or fractal structure.

Note: While all chaotic attractors are strange, not all strange attractors are chaotic.[1]

Consider a volume in phase space defined by all the initial conditions a system may have. For a dissipative system, this volume will shrink as the system evolves in time (Liouville's Theorem). If the system is sensitive to initial conditions, the trajectories of the points defining initial conditions will move apart in some directions, closer in others, but there will be a net shrinkage in volume. Ultimately, all points will lie along a fine line of zero volume. This is the strange attractor. All initial points in phase space which ultimately land on the attractor form a basin of attraction. Note: A strange attractor results if a system is sensitive to initial conditions and is not conservative.

The surfaces which the state of a chaotic system will be confined to, given time for transients to die out.

Subject: How can I join the BITNET fractal discussion?

Q19: How can I join the BITNET fractal discussion?

A19: There is a fractal discussion on BITNET that uses an automatic mail server that sends mail to a distribution list. To join it, send a message to listserv@gitvm1.bitnet with the following as text

```
SUBSCRIBE FRAC-L John Doe
```

where "John Doe" is replaced by your name.

To unsubscribe, send the message:

```
UNSUBSCRIBE FRAC-L
```

If that doesn't unsubscribe you, you can try:

```
SIGNOFF FRAC-L (GLOBAL
```

If that doesn't work or you have other problems, you can contact the list administrator. You can obtain his name by sending the message:

```
REVIEW FRAC-L
```

Subject: 3-D fractals

Q20: How can 3-D fractals be generated?

A20: A common source for 3-D fractals is to compute Julia sets with quaternions instead of complex numbers. The resulting Julia set is four dimensional. By taking a slice through the 4-D Julia set (e.g. by fixing one of the coordinates), a 3-D object is obtained. This object can then be displayed using computer graphics techniques such as ray tracing.

The papers to read on this are:

A. Norton. Generation and Display of Geometric Fractals in 3-D. *SIGGRAPH '82:* 61-67.

J. Hart, D. Sandin, and L. Kauffman. Ray Tracing Deterministic 3-D Fractals. *SIGGRAPH '89*: 289-296.

Instead of quaternions, you can of course use other functions. For instance, you could use the regular Mandelbrot formula, but vary both c and z_0, resulting in a 4-D Mandelbrot set.

Another way of generating 3-D fractals is to use 3-D iterated function systems (IFS). These are analogous to 2-D IFS, except they generate points in a 3-D space.

Subject: What are some general references?

Q21: What are some general references on fractals and chaos?

A21: Some references are:

Books:

Benoit Mandelbrot. *The Fractal Geometry of Nature*. New York: W. H. Freeman and Company. ISBN 0-7167-1186-9. In this book Mandelbrot attempts to show that reality is fractal-like. He also has pictures of many different fractals.

---

[1] Grebogi et.al. Strange Attractors that are not chaotic. *Physica* D13 (1984): 261-268.

H. O. Peitgen and P. H. Richter. *The Beauty of Fractals.* New York: Springer-Verlag (1986). ISBN 0-387-15851-0. Lots of neat pictures. There is also an appendix giving the coordinates and constants for the color plates and many of the other pictures.

H. O. Peitgen and P. H. Richter, eds. *The Science of Fractal Images.* New York: Springer-Verlag (1988). ISBN 0-387-96608-0. This book contains many color and black and white photographs, high level math, and several pseudocoded algorithms.

Michael Barnsley. *Fractals Everywhere.* Academic Press (1988). ISBN 0-12-079062-9. This is an excellent text book on fractals. This is probably the best book for learning about the math underpinning fractals. It is also a good source for new fractal types.

James Gleick. *Chaos: Making a New Science.* Penguin Books (1988).

Ian Stewart. *Does God Play Dice?* Basil Blackwell (1990).

Robert L. Devaney. *Chaos, Fractals, and Dynamics.* Addison-Wesley (1990). ISBN 0-201-23288-X. This is a very readable book. It introduces chaos, fractals, and dynamics using a combination of hands-on computer experimentation and precalculus math. Numerous full-color and black and white images convey the beauty of these mathematical ideas.

Kenneth Falconer. *Fractal Geometry.*

Robert L. Devaney. *An Introduction to Chaotic Dynamical Systems* (2nd Ed.) Addison Wesley (1989). ISBN 0-201-13046-7. This book introduces many of the basic concepts of modern dynamical systems theory and leads the reader to the point of current research in several areas. It goes into great detail on the exact structure of the logistic equation and other 1-D maps. The book is fairly mathematical, using calculus and topology.

Proceedings of the Santa Fe Institute's Complex Systems Summer School (1988). Official Title: Dan Stein, ed. *Volume I Lectures in the Sciences of Complexity.* See esp. the first article by David Campbell: Introduction to nonlinear phenomena.

R. Devaney and L. Keen. *Chaos and Fractals: The Mathematics Behind the Computer Graphics.* This book contains detailed mathematical descriptions of chaos, the Mandelbrot set, etc.

Clifford A. Pickover. *Computers, Pattern, Chaos, and Beauty.* This book contains a bunch of interesting explorations of different fractals.

Fractal Creations. This is the book describing the FractInt program.

Heinz-Otto Peitgen, Hartmut Jürgens, and Dietmar Saupe. *Fractals for the Classroom.* Four volumes: *Introduction to Fractals and Chaos; Complex Systems and the Mandelbrot Set; Strategic Activities (Volumes I and II).* This book is aimed at advanced secondary school students (but is appropriate for others too), has lots of examples, explains the math well, and gives BASIC programs.

Roger Stevens. *Fractal Programming in C.* M&T Publishing (1989). ISBN 1-55851-038-9. This is a good book for a beginner who wants to write a fractal program. Half the book is on fractal curves like the Hilbert curve and the von Koch snow flake. The other half covers the Mandelbrot, Julia, Newton, and IFS fractals.

M. Schroeder. *Fractals, Chaos, and Power Laws.* This book contains a clearly written explanation of fractal geometry, with lots of puns and word play.

Jens Feder. *Fractals.* Plenum Press (1988). This book is recommended as an introduction. It introduces fractals from geometrical ideas, covers a wide variety of topics, and covers things such as time series and R/S analysis that aren't usually considered.

Gerald A. Edgar. *Measure, Topology, and Fractal Geometry.* Springer- Verlag (1990). ISBN 0-387-97272-2. This book provides the math necessary for the study of fractal geometry. It includes the background material on metric topology and measure theory and also covers topological and fractal dimension, including the Hausdorff dimension.

H. B. Lin. *Chaos.* This is an excellent collection of papers on chaos, containing some of the most significant reports on chaosology, such as *Deterministic nonperiodic flow* by E.N. Lorenz.

Journals: *Chaos and Graphics* section in the quarterly journal _Computers and Graphics_. This contains recent work in fractals from the graphics perspective, and usually contains several exciting new ideas. *Mathematical Recreations* section in Scientific American.

Amygdala. P.O. Box 219 San Cristobal, NM 87564-0219. This is a newsletter about the Mandelbrot Set and other fractals.

Articles:

P. Blanchard. Complex Analytic Dynamics on the Riemann Sphere. *Bull. of the Amer. Math. Soc.* 11, 1 (July 1984): 85-141. This article is a detailed discussion of the mathematics of iterated complex functions. It covers most things about Julia sets of rational polynomial functions.

Subject: Acknowledgments

For their help with this file, thanks go to:

Erik Boman, Abhijit Deshmukh, Robert Drake, Gerald Edgar, Gordon Erlebacher, Duncan Foster, Murray Frank, Lamont Granquist, Arto Hoikkala, Carl Hommel, Robert Hood, Oleg Ivanov, J. Kai-Mikael, Leon Katz, Matt Kennel, Tal Kubo, Jon Leech, Tom Menten, Guy Metcalfe, Miriam Nadel, Ron Nelson, Tom Parker, Matt Perry, Michael Rolenz, Tom Scavo, Jeffrey Shallit, Rollo Silver, Bruce Stewart, Dwight Stolte, Tommy Vaske, Andrea Whitlock, Erick Wong, and others.

Special thanks to Matthew J. Bernhardt (mjb@acsu.buffalo.edu) for collecting many of the chaos definitions.

Copyright 1992 Ken Shirriff (shirriff@sprite.Berkeley.EDU)

z^z

— *Rollo Silver*

A few years ago John Corbit wrote to ask if I could supply him with the real and imaginary parts of $f(z) = z^z$. I'm printing the result here, since it may be of use to those of you interested studying fractals based on iterating exotic functions.

z^z is a bit tricky, but doable. Let's let $z = x + iy = re^{i\theta}$. We have $e^{i\theta} = \cos\theta + i\sin\theta$. The log function is defined as the inverse of the exponential function, so:

$$\log z = \log(re^{i\theta}) = \log r + \log(e^{i\theta}) = \log r + i\theta\ .$$

We then get:

$$\begin{aligned} z \cdot \log z &= (x + iy)(\log r + i\theta) \\ &= x(\log r + i\theta) + iy(\log r + i\theta) \\ &= x \cdot \log r - y\theta + i(x\theta + y \cdot \log r) \end{aligned}$$

so:

$$z^z = e^{z\log z} = e^{x \cdot \log r - y\theta} e^{i(x\theta + y \cdot \log r)}$$

and see equation [5], below, where of course

$r = \sqrt{x^2 + y^2}$, and $\theta = \operatorname{atan}\frac{y}{x}$.

The tricky part is that the log function is actually multi-valued:

$$\log z = \log r + i(\theta + 2\pi n)\ \text{, for } \textit{all} \text{ integer } n. \quad [1]$$

To see how this affects things, let $\varphi = \theta + 2\pi n$, so that [1] becomes:

$$\log z = \log r + i\varphi \quad [2]$$

[2] should actually be as in equation [6] below, so, since $\cos(x\varphi + y \cdot \log r) = \cos(x\theta + y \cdot \log r)$ and

$\sin(x\varphi + y \cdot \log r) = \sin(x\theta + y \cdot \log r)$, we have z^z as given in equation [7] below.

Thus the real and imaginary parts of z^z are:

$$\Re(z^z) = e^{x\log r - y(\theta + 2\pi n)} \cos(x\theta + y\log r) \quad [3]$$

$$\Im(z^z) = e^{x\log r - y(\theta + 2\pi n)} \sin(x\theta + y\log r) \quad [4]$$

This leaves open the tricky question of "which n to choose?" — it obviously makes a difference! The wrong choice can lead to weird "slip planes" in pictures based on this function (or other functions which are multi-valued, like $\sqrt{z}$). See for instance John Dewey Jones' picture #35 (also known as #166), which I sent out with newsletter #3. I discussed the problem in my *Bifurcations* article (Amy #3, p.1).

The proper treatment of these matters involves *Riemann surfaces,* a subject which would lead us far afield. I refer interested parties to Ahlfors' *Complex Analysis,* pages 97-99 and 229-235.

In the meantime remember that slip planes and other "interesting" artifacts of multivaluedness are not actual features of the fractal universe, but are rather flaws in the glasses through which we observe that universe!

$$z^z = e^{x \cdot \log r - y\theta} \cdot [\cos(x\theta + y \cdot \log r) + i\sin(x\theta + y \cdot \log r)] \qquad [5]$$

$$z^z = e^{x \cdot \log r - y\varphi} \cdot [\cos(x\varphi + y \cdot \log r) + i\sin(x\varphi + y \cdot \log r)] \qquad [6]$$

$$z^z = e^{x \log r - y(\theta + 2\pi n)} [\cos(x\theta + y \log r) + i\sin(x\theta + y \log r)] \qquad [7]$$

An Ingenious Dodge: Mandelbrots from Julias

From a letter by Warut Roonguthai

I'm very interested in the double/double spiral found by A.G. Davis Philip and shown in Amygdala #22, page 3. It's an essentially new view from the Mandelbrot set. I'm a Fractint enthusiast and would like to generate it for myself using this program, but the magnification (about 2.7×10^{15}) exceeds the range that Fractint can handle (which is about 1.56×10^{13}).

Here's the solution. From my experience, I've noticed that the same view as from the Mandelbrot set can be seen at much lower magnification from the corresponding Julia set. So I used the coordinate of the center of Philip's picture as the *c*-parameter of the Julia set:

$$c = -0.7485{,}8089{,}7648{,}764 + 0.0630{,}6469{,}1777{,}6402i$$

Then I zoomed in on this Julia set at the origin. Finally, I encountered the double/double spiral as expected. At a magnification of 17,000, the picture is beautiful! Although this magnification is in the range of integer math, you have to use floating-point math because the precision of the parameter exceeds integer math range. The maximum dwell is 25,000 to eliminate all points that do not escape to infinity. The escape radius is 2 and the log palette is used.

Fractals in Special Effects for Movies: an E-mail Conversation

Does anybody know how fractals are used to represent any arbitrary graphic? In other words we normally go from some mathematical formula and see the resulting graphic, I basically want to know the inverse process of going from some arbitrary graphic to the resulting math formula that encodes it as a fractal.

I believe that these details are essentially the process in the Barnsley fractal compression methodology which I don't know. However, if I am not mistaken, they have patented the procedure in which case it is now public knowledge even if you can't use the idea for free. Anybody know the essence of the idea who can lay it out for me? I have seen various fractal pictures that must have originated in this manner (with or without Barnsley compression) and would like to learn how to capture some painting or photograph as a mathematical fractal representation.

Thanks in advance for the info.
— *Henry Manaster*

Henry Manaster might find material of interest in a chapter entitled "Making Chaotic Dynamical Systems to Order" by Michael F. Barnsley in Chaotic Dynamics and Fractals by Barnsley and Demko, Academic Press 1986.

That article includes several references, most of which, unfortunately, were in press at the time of publication. Nevertheless, you may find some of them from the following partial citations:

- M. F. Barnsley, S. G. Demko, Iterated function systems and the global construction of fractals, Proc. Roy. Soc. London (in press).
- M. F. Barnsley, V. Ervin, D. Hardin, J. Lancaster, Solution of an inverse problem for fractals and other sets, Georgia Tech Reprint (1984).
- S. Demko, L. Hodges, B. Naylor, Construction of fractal objects with iterated function systems, Pro-

ceedings of SIGRAPH 1985.

- P. Diaconis, M. Shashahani, Products of random matrices and computer image generation, Stanford University reprint (1984).
- J. Hutchison, fractals and self-similarity, Indiana University Journal of Math., 30 (1981) 713-747.

I hope this gets you on the track.
— *Hal Harris*

A Review of FRACTAL PROGRAMMING IN C, by Roger T. Stevens

Chris Bezirtzoglou

Chapter 1: Introduction

This is a quick introduction to the current state of the Chaos revolution: "Monster" Curves, Strange Attractors, Iterated Function Systems, Population Curves and Bifurcation Diagrams, Mandelbrot and Julia Sets.

Chapter 2: What are Fractals?

The fundamental questions are drawn. What are Fractals and what are good for? The author's answer is that fractals appear to provide solutions to many previously unanswered questions at the frontiers of the physical sciences.

Chapter 3: Hardware Requirements

This is common sense. Try to see the beautiful colors of a fractal in a Hercules Graphics Card or do a ray-tracing for the last fractal landscape you have produced in your good old PC and you will be bored to death. So you need at least a fast 80286 or 80386 coupled with a numerical coprocessor for calculations and a color EGA or VGA monitor to display your results. As a programming language you must use a fast one, like C (either Turbo, Microsoft or Quick) with the really hard parts coded in plain or 386-specific assembly.

Chapter 4: Saving and Compressing Display Data

We are introduced to the screen saving and restoring routines both in EGA (640x350x16 colors) and VGA (320x200x256 colors). The writer has chosen to use the .PCX graphics format.

Chapter 5: Tools for Graphics Display

In this chapter we are introduced to the general (both Turbo C and Microsoft or Quick C) graphics toolbox. This toolbox has the following functions: Set Display Mode, Set EGA/VGA Palette, Clear Screen, Plot a Point on the Screen with/without using Assembly Language, Display a Byte on the Screen, Read a Pixel from the Screen, Read a Byte from a Color Plane, Draw a Line on the Screen, Fill a Triangle, Fill an Oval, Display Cursor, Move Cursor, Change Display Colors, Get Saved Screen Parameters, Turtle Graphics (Point, Turn, Step).

Chapter 6: The Lorenz and Other Strange Attractors

Playing with the first discovered strange attractor: the Lorenz attractor. Examining the attractor found from Clifford A. Pickover. Using the Runge-Kutta fourth-order method of integration.

Chapter 7: The Population Equation and Bifurcation Diagrams

The bifurcation diagram: a graph we use to describe the complex behavior of the population equation. The discovery of the universal Feigenbaum number.

Chapter 8: The Snowflake and Other von Koch Curves

An old fractal: the von Koch snowflake (the initiator is a equilateral triangle). A variation on the same subject: the Gosper curve (the initiator is a regular hexagon). Working with other von Koch curves: the 3 / 8 / 18 / 32 / 50-segment Quadric von Koch curve (the initiator is a square). Strange kinds of initiator and generator curves giving "monkey's tree" and "split snowflake halls".

Chapter 9: Peano Curves

Constructing the original (the initiator is a straight horizontal line) and the modified Peano curves. Building the original and the modified Cesaro Triangle curve. Variations on the theme: the Polya Triangle curve, the Peano-Gosper curve and the Peano 7/13-segment snowflake.

Chapter 10: The Hilbert Curve

The Hilbert curve is one of the Peano family of curves. The writer examines both the two and three dimensional version of it. An application of Hilbert curves: compressing display information.

Chapter 11: The Sierpinski Curve

The Sierpinski family of curves: the triangle, the gasket, and the box. Strange cousins of the Sierpinski triangle: implementing the chaos algorithm.

Chapter 12: Trees

This is a chapter on ecology; growing up trees. You can design any tree once you are given the following parameters: initial length and width of the stem, the left and right branching angles and the level of recursion, the cross-sectional areas of tree branches at a given height for the left and right side of the tree.

Chapter 13: Working with Circles

Working with a line that have neither a starting point neither an ending one: a circle. The author introduce us to the Apollonian circles, to the inversion technique and build for us a jewelry of astonishing beauty, the Pharaoh's Breastplate.

Chapter 14: The Mandelbrot Set

The "beast" is back. Another classical excursions to the properties of the Mandelbrot set.

Chapter 15: Julia Sets

This chapter is about Julia Sets. How to plot the Julia Set, how to make a Julia Dragon, how to use the binary decomposition technique for producing Julia sets.

Chapter 16: Dragon Curves

This is a fairy-tale about known dragon curves: The Harter-Heightway and the Twin dragons, the Self-squared and the San Marco dragons. Furthermore it examine a technique for drawing dragon outlines.

Chapter 17: Phoenix Curves

This is a chapter about the Phoenix family of curves. They have taking this name because they resemble the phoenix bird.

Chapter 18: Three-Dimensional Dragons

Using the dragon outline method, discussed in chapter 16, to draw a three-dimensional dragon. This is done due to the lack of big processor speed.

Chapter 19: Newton's Method

Explaining the Newton's method for finding equations roots. Special investigation is carried out for the $z -1=0$ and $z -2z-5=0$ formulas.

Chapter 20: Brownian Motion

What do we mean by the term Brownian motion? Plotting a moving particle first in one and then in two dimensions using a Brownian motion generator.

Chapter 21: Fractal Landscapes

Playing God. Constructing a fractal landscape. The midpoint displacement technique to produce a real-world-like terrain.

Chapter 22: Iterated Function Systems

Evaluate iterated function systems using first a deterministic and secondly a chaos algorithm.

Appendix A: Tools Library

That's the complete program code for the Graphics Tools Library (covering twenty three pages) presented by the author, originating from the author's previous book Graphics Programming in C.

Appendix B: Programs for Hercules Graphics

Adapter A massive one hundred thirty seven pages adoptions of the programs for the Hercules monochrome (really ugly) graphics adapter. The author strongly advise anyone interesting in fractals to buy a color monitor and card. Furthermore if anyone wants to get the maximum of the book and insists in using a Hercules he should write his own save_screen and restore_screen functions.

Bibliography

There is a small thirty nine reference list. Among these exists four magazine articles (Micro Cornucopia) with special interest to i386 computer users.

Conclusion

This book is definitely a must for the fractal practitioner. It is well presented, covers a lot of techniques, introduce the reader to many different kind of fractal and includes a lot of C program code. It resembles the style of Jean-Paul Delahaye book Geometric and Artistic Graphics: Design Generation with Microcomputers. There the programs are coded in BASIC, here in C (you must have in mind that this book is only one year old). In a word I highly recommend the book and advise anyone that would like to enter the full code to buy the book/disk combination, as it really worth the extra ten or fifteen dollars.

Sculpture of Quaternion Julia Sets: an E-mail Conversation

Earlier this year a series of messages appeared on the FRAC-L conference of the Bitnet about making actual fractal sculpture. I found it fascinating, and I hope most of my subscribers will, too.

Hello, all!

I have made proposals to the manufacturers of a couple of kinds of suitably designed computer-aided Rapid Prototyping technologies (Stereolithography is not one of them) to build physical models of 3-D Julia Sets I have rendered as pixel volumes.

There are only one or two kinds of computer-interfaced 3-D model builders which are properly suited to the task and there will be some custom software required to work the interface.

I think that most people on this list would find it exciting to see a physical sculpture of a Julia set calculated in the Quaternions, which was directly rendered from the computer-generated image. Am I right?

If there is anyone out there who might be materially motivated to see such a thing come to pass, or who can at least make a convincing argument why such a thing should happen (additional voices other than mine will help) please send these to me.

I will keep you posted as to my progress.

Thanks in advance!
— *Stewart Dickson*

Does "materially motivated" involve mere moral support or something more "material" in nature? I have some nice "computer sculptures" by Stewart Dickson that came with my "Mathematica" and I would be interested in seeing some of them turned into real objects.

I also would like to see the "Alexander horned sphere" rendered on a computer and in real life.
— *Dave Uberka*

This sounds like a nice application for computer-aided photolithography (where a UV laser shines on a pool of UV-curing resin. The laser "colors-in" a two-dimensional cross section of the desired solid, curing the resin for that cross section. The cured resin is then lowered a millimeter and the process is repeated with the next cross section.). IBM's CATIA CAD/CAM system supports this; I'm sure others do, too.

Although the litho equipment is expensive, the process is cheap and relatively fast. A few posts to some of the engineering and physics groups might find someone willing to render your data.
— *Jeff Nye*

About two years ago the Mathematical Intelligencer ran an article showing the AHS (Alexander horned sphere) and other mathematical structures rendered in bronze.

In the symbolic language Mathematica, the AHS starts off like this:

```
p1h=ParametricPlot3D[{Cos[t] (3 + Cos[u]), Sin[t] (3 + Cos[u]), Sin[u]}, {t, 0, 7/4 Pi}, {u, 0, 2Pi}];

p2h=ParametricPlot3D[{2+Cos[t+Pi] (3 + Cos[u]), Sin[u], Sin[t+Pi] (3 + Cos[u])}, {t, 0, 7/4 Pi}, {u, 0, 2Pi}];

Show[p1h,p2h]
```

[but how to continue it?]
— *david fowler*

I have a few reprints left of chapters from my book *Visions of the Future*. These include reprints of 3 chapters probably of interest to readers of this forum, including:

1. Fractals and Genetics in the Future by H. Joel Jeffrey

2. Current Techniques and Development of Computer Art, by Franz Szabo

3. Forging a Career as a Sculptor from a Career as Computer Programmer, by Stewart Dickson

I'd be happy to send out free reprints to researchers for scholarly purposes, until the reprints run out.

Just send me your name and address.
— *Cliff Pickover*

View-Master as a film format (16mm? 8mm?) is a bit odd, not exactly standard. I know that suppliers of 3D equipment are not exactly plentiful. Reel-3D, Culver City, CA is the only one I know about. The basis of the system is a special film cutter, which does a cookie-cutter-like (accurate to 0.001") job on your 35mm Ektachrome film. Note: this means your digital slide-shooting setup also has to be accurate to 0.001". I looked in my Reel-3D catalogue and found that they indeed have View-Master disk blanks in stock.

For stereo transparencies, I prefer the Fraka twin-35mm slide viewer. With a $5.00 adapter, it accepts a pair of 35mm slides as mounted by the processing lab. Your only major headache is getting the geometry right when you render your digital images — oh yes, and labeling your slides "left", "right" correctly.

Don't any of these electronic stereo viewer outfits support any graphics adapters on IBM PC's yet? I would think they must by now. I would also think that this kind of hardware would less expensive than some digital film recorders by now.

I have been using the U.S. Army High-Performance Computing Research Center's (AHPCRC, University of Minnesota) "Brick of Bytes" (BOB) interactive volume visualization freeware on a Silicon Graphics Personal Iris (4D/35TG — 32Mbytes) and Iris Indigo^2 Extreme (96Mbytes) to view Julia sets calculated in Quaternion complex 4-space and represented as pixel volumes using a program of my own authorship. At least in this system one can roll the object around and see the other side.

How does direct selective laser sintering in thermoplastic from pixel volume data sound as a "novel way of viewing 3D fractals"? My friends at University of Texas, Austin tell me that we might see the first examples of these objects in physical three-space by the end of the month (July 31).
— *Stewart Dickson*

Re Iterations on Clay Pots

I have just read Bart Lynch's piece in LEONARDO Electronic News 3(7) (July 15, 1993) on "SOUND AND CERAMICS". This is a very cool piece, on levels which go quite a bit deeper than he described in his article, I think.

What he is describing is an iterative procedure in which he is translating a physical, 3D object (a clay pot, thrown on a potter's wheel) into an abstract representation. He is "sounding" or "ringing" the object, recording this event and creating an alternate, spatial abstract representation of the object: A graph, presumably a frequency spectrum(?) (he did not say precisely what graph he produced on SOUND EDIT PRO). Wait a minute, on reading more closely, he says, "...that is time dependent." This is merely an amplitude envelope, then. A pity.

At any rate, then he takes the sound envelope graph and uses it as a contour template for throwing a new pot.

If he were to make frequency spectrum pictures, he would be translating the event into the complex phase plane and then we could compare the procedure to those like computing the Mandelbrot set. I think that were Bart Lynch to approach it in this way, that he could map out the divergence and convergence of his procedure; i.e., he would be finding convergent and divergent lathed shapes.

I will refer this to some experts in the field (FRAC-L) and see what they say. I'm sure the procedure could be simulated. I think the acoustical mechanics of a lathed shape are well known. The question is mainly what kind of graphs will produce convergent sound-producing shapes. I think we can fiddle with the mapping a little until we find something that can produce regions of convergence amid a field of divergent shapes. One can imagine that convergent shapes could be analogous to some kind of purity or "music of the spheres".

Please let me know your response to these ideas. I will compile all responses I receive.

— *Stewart Dickson*

Do you have access to "LEONARDO Electronic News"? If not, I can e-mail you the article in its entirety. Oh, it's not too long, please find it below.
— *Stewart Dickson*

SOUND AND CERAMICS

— *Bart Lynch*

From Leonardo, Volume 3: Number 7:

In architecture, natural harmonies occur in Renaissance structures.

Harmonic relations of form and space were often based on the golden section and the ratios therein. These same ratios occur in the growth patterns of flowers, fish, and other components of nature. I am currently concerned with understanding why these ratios occur and why they are pleasing to us.

I have been translating sounds to three dimensional pottery using several computer programs in order to see if pleasing sounds make pleasing pottery and vice versa. [1] Using the sound program SOUND EDIT PRO, I can get a visual representation of a sound that is time dependent. That visual is saved as a picture and imported to SWIVEL 3-D where the sound form can be lathed to resemble pottery and used as a template to create actual ceramic works. Using these programs, I have also been animating the figures so that on the computer screen, the pottery forms dialogue with the sounds that created them. I see these processes as data gathering exercises that help me to understand the nature of the harmonic relations so that I will be able to use them more effectively in the future.

[1] This work was developed at Deep Creek School in conceptual collaboration with Dan Collins.

Published by Leonardo, the International Society for the Arts, Sciences and Technology (ISAST), 672 South Van Ness Avenue, San Francisco, CA 94110 USA

End of Leonardo Electronic News 3(7)

A letter from Stewart Dickson, about Julia Set Sculpture

Dear Mr. Silver:

Thank you for your note of August 24. I have some information you may be interested in. It may not yet be in a form you can publish immediately, but I am hoping that with a little back-and-forth, we might have something to announce soon.

I have made some progress in finding what could be the only engineering solution to directly building deterministic fractals using computer-aided prototyping technologies. I can send you some background material on this if you like. Let me know.

The Selective Laser Sintering process has the ability to build on a particle-by-particle basis a computer-specified object without the requirement for ancillary supporting structures. That is, the building process ensures that the object is built fully-supported, except that there is no built-in countermeasure for material deformation (warping) during the building process.

The Selective Laser Sintering technology permits building an object as if it were a pixel volume treated as a series of horizontally-oriented one-bit-per-pixel image slices. The only requirement to obtaining a solid final object is some abstract verification that the pixel volume is connected. Otherwise, there is no technical barrier preventing the building of a disconnected or fractal volume.

The Selective Laser Sintering process was invented by Dr. Joseph Beaman, Professor, Department of Mechanical Engineering, University of Texas at Austin.

The process has been commercially licensed and developed by the DTM Corporation, Austin, Texas, where Professor Beaman holds a position of Directorship. The DTM Corporation is a partially-owned subsidiary of B. F. Goodrich Corporation, Cleveland, Ohio.

I have calculated Julia sets in Quaternion complex 4-space using a program of my own authorship. These Julia sets I have represented as pixel volumes in two forms — 8-bits-per-voxel, with a 256-entry color map and 1 bit-per-voxel — at numerous cubic dimensions. I have made some volume renderings of these data sets and have presented these images in 35mm slides I have directly recorded on Ektachrome film using a Management Graphics, Inc. Solitaire 8xp digital image recorder. Software I have used to make the volume renderings has been the University of Minnesota Army High-Performance Computing Research Center (AHPCRC) "Brick-of-Bytes" (BOB) public-domain software on a Silicon Graphics Personal Iris and Iris Indigo^2. I have also used Pixar ChapVolumes on a Pixar Image Computer.

Please let me know if you would have a use for slides of the computer renderings I have done. I have a set of approximately 13, which fairly clearly illustrate my proposal. I could send a set to you, but not without some time and expense.

As it stands, the commercial machines, marketed and operated by DTM Corp. have a larger build envelope (a cylindrical diameter 12" by 14" axial) and build to higher precision than do the research machines at UT Austin, which build to roughly a 4" cube. The research machines, however, are open systems which routinely use custom software. The commercial machines only accept a data interface in the form of the so-called 'STL' object file format, an extremely bulky polygon-based specification. UT Austin has defined a so-called "toggle" interface which is very nearly a one-bit-per-voxel run-length-encoding of laser scan-line specifications. I have calculated a nearly 25:1 data compression using the UT toggle file compared to the best that could be done with 'STL'.

To date I have delivered nine data sets to [several people].

The data sets are pixel volumes representing three different iterative procedures:

$$z \rightarrow z^2 + [-0.1103 - 0.67037i + 0j + 0k] \ ,$$

$$z \rightarrow e^{\frac{i\pi}{2}} z^2 + e^{-\frac{i\pi}{2}} [-0.745 + 0.113i + 0.01j + 0.01k] \ ,$$

and

$$z \rightarrow e^{\frac{i\pi}{2}} z^2 + e^{-\frac{i\pi}{2}} [0.2809 - 0.53i + 0j + 0k] \ ,$$

where z is a complex quaternion four-vector of the form $[a + bi + cj + dk]$. The pixel volume represents the three-space spanned by $[1 + 1i + 1j + 0k]$.

(After V. A. Norton)

The pixel volumes all are calculated over a domain of bounds (+-[1.0, 2.0], +-[1.0, 2.0], +-[1.0, 2.0]), are strictly cubic and centered at the origin. Some variations in the bounds of calculation are made to better fit the set in the calculated volume. All data sets are specified to be reconstructed at a fixed parameter of 0.01" per voxel in all (x, y, z) directions. I have delivered Julia sets for each of the above three procedures at each of three voxel dimensions: 64 x 64 x 64, 200 x 200 x 200, 400 x 400 x 400 in "toggle" data file format. In other words, the entire domain is sampled (each procedure is calculated) for these cubic grids. The sizes of the resulting physical objects will then be slightly less than 0.64" x 0.64" x 0.64", 2" x 2" x 2" and 4" x 4" x 4", respectively.

Thus, the data sets I have delivered represent three-step "fractal zooms" on three different procedures. The process can be extended up to at least 800 x 800 x 800 voxel resolution, if the toggle-format interface ever becomes available on the larger, commercial SLS machines. Further, the same maximum resolution can be calculated on ever smaller regions of the domain over which the Julia set is defined, resulting in the resolution of ever finer detail. As you know, the process is bounded only by the floating-point numerical resolution of the machine on which the calculation is done.

To date, activity has occurred on approximately June 21, when a graduate student named Benny Wu attempted to build the data set at 400 x 400 x 400 resolution of the second procedure listed above on an experimental machine which builds directly in bronze powder. The attempt failed due to warping of the solidified material. Benny made another attempt on a Polycarbonate machine, I think, just a week or two ago. This attempt reportedly failed due to some unspecified software problem. I told Professor Crawford to let me know if the problem was on my end or their end and I have heard nothing since.

My progress has been a little slow. I have been working on the relationship with UT Austin since approximately January. I have a letter of agreement from Rich Crawford, outlining the proprietary value of the object data, agreeing to credit each other in all presentations and so on. I have had some problems in the past with polygon model data I have supplied to DTM Corporation getting out of control, the resulting objects at least being passed around freely with no credit given. I would greatly appreciate any support you can give in principle for the project I am pursuing. Please feel free to contact any of the people I have listed in connection with this project, if you wish. Thank you.

Very truly yours,

Stewart Dickson

Barnsley's Compression Software: A review

Hal Harris, Department of Chemistry, University of Missouri — St. Louis

I don't believe it has been mentioned on the [FRAC-L] list that there is a nice review of Iterated Systems' software package called "Poem Colorbox" in *Essential Guide to Windows,* a special issue of Byte magazine, recently published. "Poem Colorbox" is a DOS/Windows package that compresses images by as much as 30X "without loss of resolution" using fractal compression methods (IFF). The whole package comes on one disk and costs $500. The review was quite favorable. The major negative was the time required. The reviewer, Steve Apiki, tried a compression of a 24-bit image approximately 750 KB in size, using a 25 MHz 386DX with a 4 MB swap file. A "fast" 9:1 compression took 3.5 minutes and gave "very good quality". A 29:1 compression took about 42 minutes. For C or Visual Basic developers who want to deliver images without using lots of space, it looks like this product would be a good choice.

Further, it is an answer to the question "Are Fractals good for anything?"

The review is found on page 95 of the issue.

Mazes for the Mind: Computers and the Unexpected

by Cliff Pickover

This review came out before *Mazes for the Mind* became available in November. I feel that it is still of considerable interest to many Amygdala subscribers. — *Editor.*

Clifford A. Pickover's new book takes the reader on a roller-coaster ride through an unpredictable and exciting universe of computers, games, puzzles, mazes, and computer art. In addition to games of his own devising, Pickover offers puzzles inspired by those dating back several centuries, from various civilizations — some of which are deceptively easy and others fiendishly difficult. None require specialized knowledge, and all are designed to stimulate creativity.

Sample chapter titles:

Fractals and Feminism, My Computer Esophagus, The Cro-Magnon Conquest Game, Electronic Ant Farms, Fantastic Feather Fractals, Music beyond Imagination, Strange Technology, Weird Numbers, Fractal Spiders, Computers and Near-Death Experiences.

Here are some quotes:

"There seems to be no end to the mathematical and mental riches Clifford Pickover keeps giving us. In this, his latest book, the central theme is exploration, with emphasis on the computer as a recreational tool. The book is a feast of puzzles, science fiction, weird numbers, curious sequences, strange mazes and games, hyperdimensional structures, fractals, chaos, unorthodox chess and music, computer lava lamps, pi curios, games, and a thousand other points of mathematical light — all interlaced with dazzling illustrations." — Martin Gardner, *Scientific American*

"A delightful trip along the fractal frontier between art and mathematics. Once again, it raises the old problem: does mathematics really exist, or do we make it all up? If so, there are some pretty weird minds out there — and good luck to them!" — Arthur C. Clarke

"Buckle your mental seatbelts, and keep your eyes open: it's a whole new ride on Pickover's visual express." — A. K. Dewdney, *Scientific American*

"Join this marvelous computer safari — experience the kingdom of the slugs and fractal ants, learn of extraterrestrial messages in our genes and of music machines, discover fascinating new computer worlds in art, music, mathematics, and science." — Theoni Pappas, author of *The Joy of Mathematics*

"Clifford Pickover is the best new math writer in years. His work coruscates with wit and energy. Buy this book and feed your head!" — Rudy Rucker, author of *The Fourth Dimension and Infinity and The Mind*

"Pickover's dazzling array of tortuous mind-benders and arcane minutiae delights and surprises. It's easy to get trapped in his enticing labyrinth of seductive mental games." — Ivars Peterson, *Science News,* author of *The Mathematical Tourist.*

The Fractal Transform

Michael Barnsley and Louisa Anson

This book is a sequel to the Michael Barnsley's extremely successful book, *Fractals Everywhere.* Without assuming a great deal of technical knowledge, the authors explain the workings of the Fractal Transform: a compression tool for storing high-quality images on a computer using a minimal amount of space. Barnsley uses examples and algorithms to explain how to transform a stored pixel image into its fractal representation.

Author

Ken Shirriff is currently a Ph.D. graduate student in operating systems at the University of California, Berkeley. He is interested in the mathematical analysis of fractal structure, was responsible for the Unix port of the Fractint program, and is active on the Usenet newsgroup sci.fractals. He can be reached by electronic mail at:

shirriff@cs.berkeley.edu

Amygdala is published by Rollo Silver every eight to twelve weeks.

Address letters, comments, subscription orders, to: *Amygdala; Box 219; San Cristobal, NM 87564; USA.*

Email: rsilver@lanl.govPhone: 505/586-0197

Compuserve: 71174,1453

This issue was produced using FrameMaker on a Macintosh IIcx computer.

Book Reviews Wanted

If you run across a book on the subject of fractals, etc. that seems particularly illuminating, and has not yet been reviewed in Amygdala, think about writing a review of the book and sending it in for possible publication in Amygdala.

Submitting Articles

Here are some guidelines for submitting articles for publication in Amygdala.

1. Type your article as you would like to see it appear in the newsletter. Please do not send handwritten drafts!

Send it (*my* order of preference): (A) On 3.5" diskette in Macintosh format for FrameMaker, MacWrite, WriteNow, or MS Word; or as a text file. Please also enclose paper copy so I can see your intent. (B) On 3.5" diskette in IBM format, text file. I have no way to deal with 5.25" diskettes. (C) Paper copy.

2. Illustrations should be either greyscale (suitable for halftoning) or black/white; not color! (A) Normally, illustrations will be printed in full column width, so you should make them 3.25" wide, if possible — provided that they're 300 dpi resolution. If they're grainier, make them larger if possible, so that they'll look good when reduced to 300 dpi. (B) Make sure that you clearly indicate which illustrations go where in the text! (C) All in all, it's better not to have captions welded into your pictures. Let me put them in ad lib. (D) I can handle illustrations on diskette in MacPaint, MacDraw, CricketDraw, Photoshop, or Adobe Illustrator formats.

3. Please send along a short biographical note, which I will try to publish in the same issue as your article.

4. Please include your telephone number, in case I have to reach you in a hurry with questions.

Circulation

As of August 27, 1994, Amygdala has 480 subscribers, 153 of whom have the supplemental color slide subscription.